An Autoregressive Recurrent Mixture density Network For Parametric Speech Synthesis

AN AUTOREGRESSIVE RECURRENT MIXTURE DENSITY NETWORK FOR PARAMETRIC SPEECH SYNTHESIS

Xin Wang[1,2]*, Shinji Takaki*[1]*, Junichi Yamagishi*[1,2,3]

[1]National Institute of Informatics, Japan
[2]The Graduate University for Advanced Studies, Japan
[3]The Centre for Speech Technology Research, The University of Edinburgh, U.K.
wangxin@nii.ac.jp, takaki@nii.ac.jp, jyamagis@nii.ac.jp

ABSTRACT

Neural-network-based generative models, such as mixture density networks, are potential solutions for speech synthesis. In this paper we follow this path and propose a recurrent mixture density network that incorporates a trainable autoregressive model. An advantage of incorporating an autoregressive model is that the time dependency within acoustic feature trajectories can be modeled without using conventional dynamic features. More interestingly, experiments show that this autoregressive model learns to be a filter that emphasizes the high frequency components of the target acoustic feature trajectories in the training stage. In the synthesis stage, it boosts the low frequency components of the generated feature trajectories and hence increases their global variance. Experimental results show that the proposed model achieved higher likelihood on the training data and generated speech with better quality than other models when dynamic features were not utilized in any model.

Index Terms— Speech synthesis, Autoregressive model, Mixture density network, Recurrent neural network

1. INTRODUCTION

Parametric speech synthesis aims at generating speech based on textual information. It uses an acoustic model, e.g., a hidden Markov model (HMM), to generate the acoustic features from textual information and then a vocoder to convert the generated features into the speech waveform [1]. Recently, various neural-networks-based models have been proposed to better map the textual features into acoustic ones [2, 3]. There are also neural networks that directly model spectral features to avoid artifacts caused by vocoders [4, 5].

This paper proposes a recurrent mixture density network that incorporates an autoregressive model. This proposed model first uses a recurrent neural network (RNN) to extract a hidden representation from the input linguistic feature. Then, it transforms the output of the RNN as the model parameter of a mixture density network (MDN) that depicts the distribution of the target acoustic features [6]. This RNN-based MDN is called the recurrent mixture density network (RMDN) in this paper. Based on the RMDN, the proposed model further assumes that the mean of the target feature distribution is dependent on the observed features of previous steps, and this cross-time dependency can be modelled using an autoregressive (AR) model [7]. The entire model, including the AR model and RMDN, can be trained using the back-propagation algorithm with the maximum likelihood criterion.

By modeling the cross-time dependency using an AR model, the proposed model does not require the conventional dynamic features nor the classical speech parameter generation method [1]. Therefore, it is similar to the Autoregressive HMM for speech synthesis [8, 9]. However, experiments show more interesting results on the proposed model. Specifically, the AR model in the training stage learns to be a filter that emphasizes high frequency components of the training feature trajectories while attenuates their low frequency components, thus, increasing the entire model's likelihood on the training data. In the synthesis stage, the trained AR model compensates the low frequency components of the generated trajectories and increases their global variance (GV) [10]. Results of subjective evaluation show that the proposed model is better than the RNN and plain RMDN when dynamic features are not used in any model.

In Section 2, we introduce the RNN and MDN. In Section 3, we present the proposed model. Then, we show the experiments in Section 4, and discuss the future work and draw the conclusion in Section 5 and Section 6, respectively.

2. NEURAL NETWORKS FOR SPEECH SYNTHESIS

2.1. Recurrent Neural Network

The basic task of a hidden layer in an RNN is to transform the input $\boldsymbol{x}_t$ and previously extracted hidden state $\boldsymbol{h}_{t-1}$ into a new vector $\boldsymbol{h}_t$:

$$\boldsymbol{h}_t = \mathcal{F}(\boldsymbol{W}_{ii}\boldsymbol{x}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i). \quad (1)$$

Here, $\mathcal{F}$ is a non-linear function; $\boldsymbol{W}_*$ is the transformation matrix and $\boldsymbol{b}$ is the bias. A vanilla RNN based on this operation is difficult to train because of the gradient vanishing and exploding problem [11]. As a solution, the long short term memory (LSTM) unit, where trainable gates control the input, output and the state of the memory cell [12], has been proposed to replace the simple $\mathcal{F}(.)$.

For speech synthesis, an RNN network with LSTM units and deep bi-directional time dependency (DBLSTM-RNN) has been reported [3]. In this system, the RNN derives $\boldsymbol{h}_t$ based on $\boldsymbol{x}_t$ that encodes the textual information. Then, it transforms $\boldsymbol{h}_t$ into the acoustic feature vector $\boldsymbol{o}_t$, based on which speech can be constructed.

2.2. Mixture Density Network

Different from neural networks that only give a point estimation for the target data, the network in an MDN predicts the value of a parameter set based on which probability density function (PDF) of the target variable can be specified [6]. An MDN may use the Gaussian mixture model (GMM) as the PDF of the acoustic feature vector $\boldsymbol{o_t}$

$$p(\boldsymbol{o}_t; \mathcal{M}_t) = \sum_{m=1}^{M} \omega_t^m \mathcal{N}(\boldsymbol{o}_t; \boldsymbol{\mu}_t^m, \boldsymbol{\Sigma}_t^m). \quad (2)$$

In Equation 2, M is the number of mixture components and $\mathcal{M}_t = \{\omega_t^1, \cdots, \omega_t^M, \boldsymbol{\mu}_t^1, \cdots, \boldsymbol{\mu}_t^M, \boldsymbol{\Sigma}_t^1, \cdots, \boldsymbol{\Sigma}_t^M\}$ is the parameter set whose value is predicted using the neural network that takes $\boldsymbol{x}_t$ as input. For simplicity, the dependency of $\mathcal{M}_t$ on $\boldsymbol{x}_t$ is omitted in this paper. Also note that $\boldsymbol{\Sigma}_t^m$ is usually assumed to be diagonal [6].

As a generative model, an MDN can be trained using the back-propagation algorithm under the maximum likelihood criterion. For speech synthesis, an MDN based on a feedforward neural network has been proposed [13] for speech synthesis. An MDN can also be constructed based on an RNN, which results in an RMDN.

3. AUTOREGRESSIVE RECURRENT MIXTURE DENSITY NETWORK

3.1. Definition

Our proposed model is based on the RMDN, yet it is assumed that the observations $\boldsymbol{o}_{t-K:t-1} = [\boldsymbol{o}_{t-K}^\top, \cdots, \boldsymbol{o}_{t-1}^\top]^\top$ in the past K time steps affect the mean value of the GMM at the current time t. Accordingly, it defines the PDF of the target feature as

$$p(\boldsymbol{o}_t|\boldsymbol{o}_{t-K:t-1}; \mathcal{M}_t) = \sum_{m=1}^{M} \omega_t^m \mathcal{N}(\boldsymbol{o}_t; \boldsymbol{\mu}_t^m + f(\boldsymbol{o}_{t-K:t-1}), \boldsymbol{\Sigma}_t^m). \tag{3}$$

Given $\boldsymbol{o}_t \in \mathbb{R}^D$, the function $f(\boldsymbol{o}_{t-K:t-1}) : \mathbb{R}^D \to \mathbb{R}^D$ summarizes the past observation and changes the mean of each mixture. Among various parametric forms of $f(\boldsymbol{o}_{t-K:t-1})$, the proposed model sets

$$f(\boldsymbol{o}_{t-K:t-1}) = \sum_{k=1}^{K} \boldsymbol{a}_k \odot \boldsymbol{o}_{t-k} + \boldsymbol{b}, \tag{4}$$

where $\boldsymbol{b} \in \mathbb{R}^D$ is the bias and $\boldsymbol{a}_k \in \mathbb{R}^D$ is a vector that scales $\boldsymbol{o}_{t-k}$ by element-wise production $\odot$. Note that $\boldsymbol{o}_t = \boldsymbol{0}, \forall t \in (-\infty, 0]$.

As the proposed model defines a PDF similar to that of the Autoregressive HMM [8], we call it as Autoregressive RMDN (AR-RMDN). An AR-RMDN with K=2 is plotted in Figure 1. The context-dependent $\mathcal{M}_t$ in the AR-RMDN is predicted by the RNN that takes linguistic feature vectors as input. Although $\boldsymbol{a}_k$ and $\boldsymbol{b}$ can also be context-dependent, our experiments suggest that a better approach is to set $\boldsymbol{a}_k$ and $\boldsymbol{b}$ as context-independent (or time-invariant). In the training stage, the weights of RNN and $\{\boldsymbol{a}_k, \boldsymbol{b}\}$ can be trained using the back-propagation algorithm under the maximum likelihood criterion.

3.2. Interpretation

As the AR-RMDN uses the AR model to depict the cross-time dependency of the target feature, it is similar to the Autoregressive HMM [8]. Compared with the HMM-based or neural-network-based models using dynamic features or trajectory models [14, 15], the AR model assumes the distribution of the current time step is affected only by the past observations, which makes the model both efficient in computation and consistent in the statistical scense [8].

Instead of only explaining the AR-RMDN as a probabilistic model [16], we interpret it further based on the signal and filter theory. Given training data $\boldsymbol{o}_{1:T} = [\boldsymbol{o}_1^\top, \cdots, \boldsymbol{o}_T^\top]^\top$, where $\boldsymbol{o}_t \in \mathbb{R}^D, \forall t \in [1, T]$, the model's likelihood can be calculated as

$$p(\boldsymbol{o}_{1:T}; \mathcal{M}_{1:T}) = \prod_{t=1}^{T} p(\boldsymbol{o}_t|\boldsymbol{o}_{t-K:t-1}; \mathcal{M}_t) = \prod_{t=1}^{T} p(\boldsymbol{c}_t; \mathcal{M}_t), \tag{5}$$

where

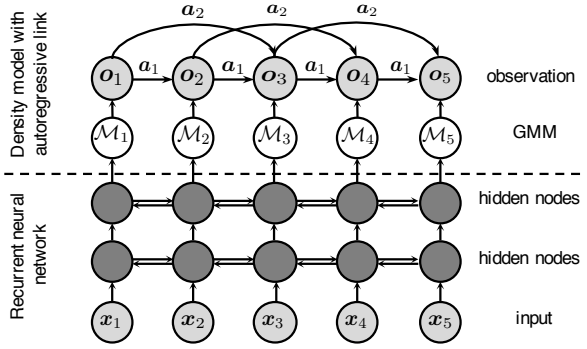

Fig. 1: Autoregressive recurrent mixture density network ($K = 2$).

$$p(\boldsymbol{c}_t; \mathcal{M}_t) = \sum_{m=1}^{M} w_t^m \prod_{d=1}^{D} \frac{1}{\sqrt{2\pi\sigma_{t,d}^{m2}}} \exp(-\frac{c_{t,d} - \mu_{t,d}^m - b_d}{2\sigma_{t,d}^{m\,2}}), \tag{6}$$

and

$$c_{t,d} = o_{t,d} - \sum_{k=1}^{K} a_{k,d} o_{t-k,d}. \tag{7}$$

Here, $o_{t,d}, \mu_{t,d}^m, a_{k,d}$, and b_d represent the d-th order element of $\boldsymbol{o}_d, \boldsymbol{\mu}_t^m, \boldsymbol{a}_k$, and $\boldsymbol{b}$, respectively, and $\sigma_{t,d}^m$ is the d-th element of the diagonal $\boldsymbol{\Sigma}_t^m$. The equations show that the model's likelihood can be calculated over a bundle of signals $\boldsymbol{c}_{1:T,d} = [c_{1,d}, \cdots, c_{T,d}]^\top, d \in [1, D]$ and each individual signal $\boldsymbol{c}_{1:T,d}$ is the output of a filter that takes $\boldsymbol{o}_{1:T,d} = [o_{1,d}, \cdots, o_{T,d}]^\top$ as input. The filter for the d-th signal can be written in the z-domain as

$$A^d(z) = 1 - \sum_{k=1}^{K} a_{k,d} z^{-k}. \tag{8}$$

In the synthesis stage, acoustic features are generated as $\hat{\boldsymbol{o}}_{1:T} = \arg\max_{\boldsymbol{o}_{1:T}} p(\boldsymbol{o}_{1:T}; \mathcal{M}_{1:T})$. If only the mixture m_t^* with the largest weight is used for generation at time t, it can be shown that $\hat{\boldsymbol{o}}_{1:T,d} = [\hat{o}_{1,d}, \cdots, \hat{o}_{T,d}]^\top$ is equivalent to the output of an AR model $H^d(z) = \frac{1}{A^d(z)}$ whose input $\hat{\boldsymbol{c}}_{1:T,d} = [\mu_{1,d}^{m_1^*}, \cdots, \mu_{T,d}^{m_T^*}]^\top$ is predicted by the RMDN part of the AR-RMDN.

An RNN with a recurrent output layer [17] is equivalent to an RMDN with $M = 1$, $\boldsymbol{\Sigma} = diag(1, \cdots, 1)$, and $\boldsymbol{\mu}_t = \boldsymbol{h}_t + \boldsymbol{W}\boldsymbol{\mu}_{t-1} + \boldsymbol{b}$, where $\boldsymbol{h}_t$ is given by the hidden layer of the RNN. Although $\boldsymbol{\mu}_t$ is affected by $\boldsymbol{\mu}_{t-1}$, this particular RNN still factorizes $p(\boldsymbol{o}_{1:T}; \mathcal{M}_{1:T}) = \prod_{t=1}^{T} p(\boldsymbol{o}_t; \mathcal{M}_t)$, which is different from the AR-RMDN's conditional PDF in Equation 5.

3.3. Implementation

The implementation of the AR-RMDN should ensure that all the learned filters $H^d(z), d \in [1, D]$ are stable. A method to ensure the stability of Autoregressive HMM is reported in [9]. Another simple strategy, used for the AR-RMDN, is to rewrite the $H^d(z)$ as a cascade form

$$H^d(z) = \frac{1}{1 - \sum_{k=1}^{K} a_{k,d} z^{-k}} = \prod_{k=1}^{K} \frac{1}{1 - \tanh(\alpha_k^d) z^{-1}}, \tag{9}$$

where the $tanh$ function ensures that the poles of $H^d(Z)$ are located within [-1,1] on the real axis. These poles can be converted into a_k^d easily, e.g., $a_1 = \tanh(\alpha_1) + \tanh(\alpha_2)$ and $a_2 = -\tanh(\alpha_1)\tanh(\alpha_2)$ when $K = 2$. In the training stage, each α_k^d can be learned based on the chain rule $\frac{\partial \mathcal{L}_\mathcal{M}}{\partial \alpha_k} = \sum_{l=1}^{K} \frac{\partial \mathcal{L}_\mathcal{M}}{\partial a_l} \frac{\partial a_l}{\partial \alpha_k}$, where $\mathcal{L}_\mathcal{M} = -\log p(\boldsymbol{o}_{1:T}; \mathcal{M}_{1:T})$.

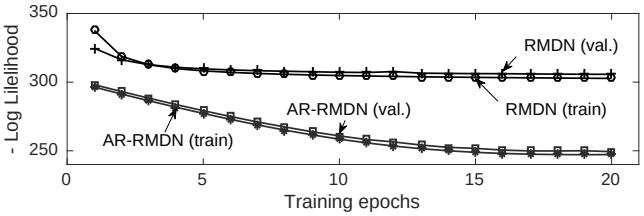

Fig. 2: Negative log likelihood of RMDN and AR−RMDN.

Table 1: Experimental systems. The recurrent layers in all systems are bi-directional and use LSTM units (DBLSTM-RNN layers).

Notations	Type of neural network	Dynamic feature
RNN	RNN	w/o
RNN+MLPG	RNN	with
RMDN	RMDN	w/o
AR−RMDN	AR-RMDN	w/o

4. EXPERIMENTS AND RESULTS

4.1. Corpus and Systems

Experiments used the Blizzard Challenge 2011 Nancy corpus that has 12072 English utterances [18]. Both the test and validation set contained 500 randomly selected utterances. Mel-generalized cepstral coefficients (MGCs) of order 60, continuous F0 trajectory, voiced/unvoiced (V/U) condition, and band aperiodicity (BAP) of order 25 were extracted for each speech frame by using the STRAIGHT vocoder [19]. The Flite toolkit [20] conducted the text-analysis for the entire corpus. The output of Flite were converted into a vector of order 382 as the input $\boldsymbol{x}_t$ to the neural network.

Experimental systems are listed in Table 1. Similar to the configuration in [3], RNN and RNN+MLPG included 2 feedforward layers with 512 nodes, 2 DBLSTM layers with 256 nodes, and a linear output layer. RMDN and AR−RMDN contained the same hidden RNN part, yet they used the MDN as the output layer. For RMDN and AR−RMDN, the MDN layer included a binomial distribution for the V/U condition, three GMMs for the MGC, F0, and BAP with the number of mixture components as $2, 2, 1$, respectively. Using two mixture components instead of a single one for MGC and F0 is supported by our prior test. AR−RMDN set the autoregressive parameter $K = 1$ for MGC, $K = 2$ for F0 and $K = 0$ for BAP.

The RNN and RNN+MLPG systems were trained first. Then, the weights of RNN, except the output layer, were used to initialize RMDN. Finally, the weights of AR−RMDN were initialized using RMDN, and $\{\boldsymbol{a}_k, \boldsymbol{b}\}$ was initialized as zero. In the generation stage, RNN+MLPG used the maximum likelihood parameter generation (MLPG) method [1] while RNN directly output the trajectories. RMDN and AR−RMDN used the mean of the most probable mixture component as the output of each frame. [1]

4.2. Analysis of Trained Model

As Figure 2 shows, the likelihood of AR−RMDN is higher than RMDN. From the perspective of model assumption, AR−RMDN is expected to be better as it takes the cross-time dependency into consideration. From the perspective of the signal and filter, AR−RMDN's performance may be due to the filtering conducted by $A(z)$. For the spectral part, the trajectory of the d-th order MGC, where $d \in [1, 60]$, is

[1]The toolkit modified based on CURRENNT [21], implementation details of AR−RMDN and speech samples can be found at http://tonywangx.github.io

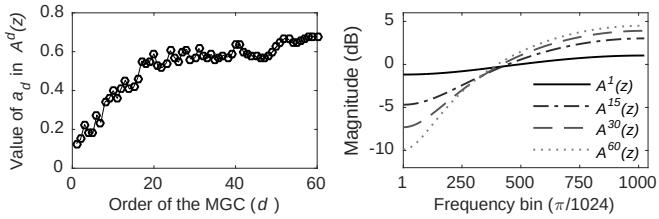

Fig. 3: $A(z)$ filter for MGC in trained AR−RMDN. Left figure shows a_d in $A^d(z) = 1 - a_d z^{-1}$, where $d \in [1, 60]$; right figure shows frequency response of $A^d(z)$ for $d \in \{1, 15, 30, 60\}$.

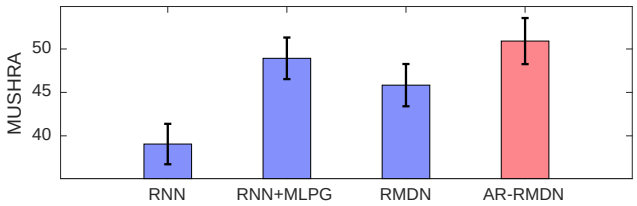

Fig. 5: Results of a MUSHRA test. Natural speech was included as reference (as well as hidden reference).

processed by $A^d(z) = 1 - a_d z^{-1}$. As Figure 3 shows, $a_d \in (0, 1)$ for $d \in [1, 60]$, and $A^d(z)$ emphasizes the high-frequency components of the natural feature trajectory while attenuates the low-frequency components. Particularly, a_d is closer to 1.0 when d is closer to 60, which means the filter for the higher order MGC trajectory emphasizes the high frequency part more severely. As the natural MGC trajectory has higher energy in the low-frequency part than in the high-frequency part, what $A^d(z)$ performs on this trajectory is similar to the whitening process conducted by a so-called inverse filter on the speech waveform [22]. However, the output of $A^d(z)$ is not the white noise but a feature trajectory that can be generated by the neural network given the input textual information. Similar results can be observed in F0 modeling where the second-order filter $A(z)$ tries to 'whiten' the spectrum of the F0 trajectory.

4.3. Analysis of Generated Feature Trajectories

In the synthesis stage, $H(z) = \frac{1}{A(z)}$ in AR−RMDN compensates the low-frequency components of the feature trajectories generated by the RMDN part and de-emphasizes their high-frequency components. The generated MGC trajectories of one sample utterance are shown in Figure 4. Interestingly, while all systems failed to generate the high frequency change in the trajectory of the 30th-order MGC, AR−RMDN generated a trajectory that had a similar dynamic range as the natural one. On the F0, AR−RMDN also generated more dynamic trajectory, e.g. a more accurate high pitch accent near the 120th frame than other systems.

For further analysis, GV is used to measure the dynamic range of generated feature trajectories. The results are shown in Figure 6. For low order MGCs, all systems generated trajectories with a similar GV to the natural data. However, for the high order MGCs, only AR−RMDN maintained the GV level of the generated trajectories. Similarly, AR−RMDN generated the F0 with larger GV than the other systems. Because the local maximum and minimum of the F0 trajectory is crucial in realization of the pitch accent [23], the over-smoothed F0 trajectory may fail to convey the pitch accent specified by the input of the neural network. The interesting point is that, instead of explicitly incorporating the GV criterion in the synthesis stage, AR−RMDN increased the GV of the generated trajectories using a filter $\frac{1}{A(z)}$, which is learned from the training data.

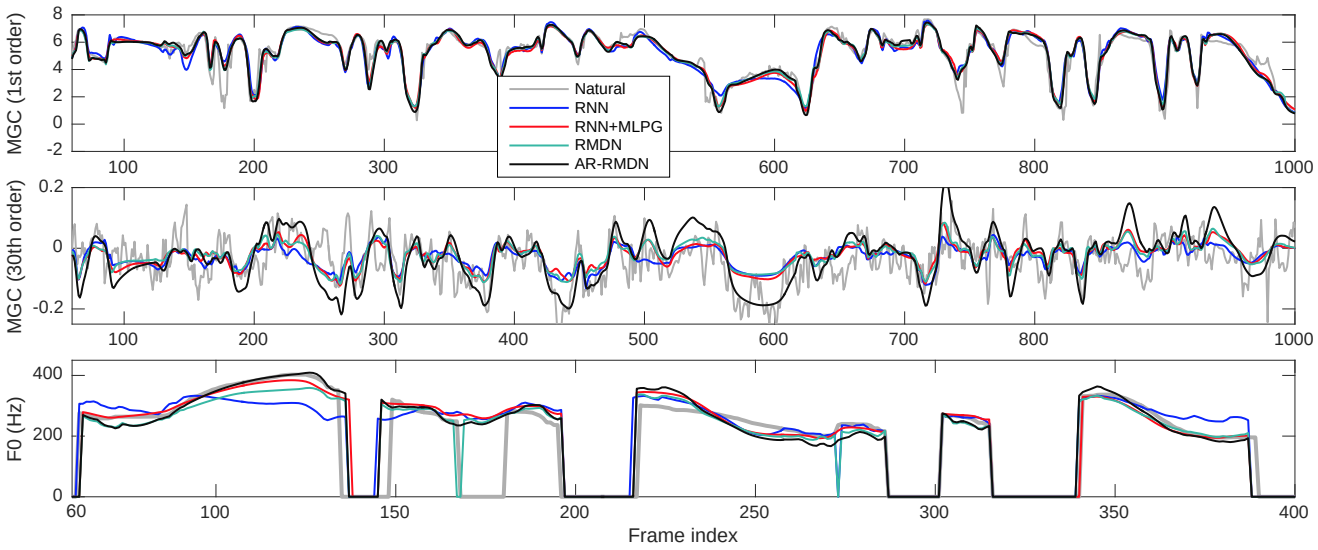

Fig. 4: Predicted MGC (1st and 30th order) and F0 trajectories.

A subjective evaluation based on a MUSHRA test with ten paid, native English listeners was conducted at the University of Edinburgh. The results are shown in Figure 5. Although statistically insignificant, AR−RMDN achieved a higher score than RNN+MLPG ($p = 0.35$ in t-test) even though AR−RMDN didn't use dynamic features or MLPG for trajectory generation. What's more, while the difference between RNN+MLPG and RMDN is not significant ($p = 0.135$), AR−RMDN is significantly better than RMDN ($p = 0.02$) and RNN ($p = 0.00$). Although AR−RMDN alleviates the over-smoothing problem, it cannot be significantly better than RNN+MLPG because the generated feature trajectories, particularly the F0, may not be smooth enough since the AR model only models the cross-time dependency locally. The discontinuity of the generated F0 by AR−RMDN can be observed near the 80th frame in Figure 4. At last, RMDN's better performance than RNN is related to the use of two mixture components for MGC and F0 since two mixture components may be robust to the outliers in the data [24, 25].

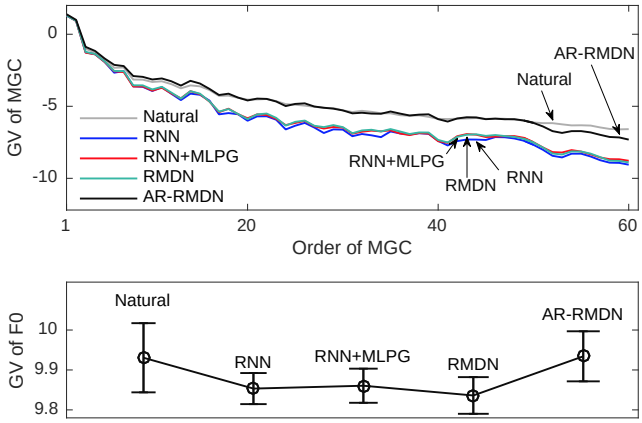

Fig. 6: Global variance of generated MGC and F0

5. DISCUSSION

As the proposed model increased the GV of the generated feature trajectories, we also compared it with another RMDN system with a postfilter on the modulation spectrum (MS) [26]. The results showed that both methods increased the GV and generated synthetic speech better than the plain RMDN. The difference is that the MS postfilter manipulates the feature trajectory in the modulation spectrum domain while the proposed model works in 'time' domain.

Although the AR-RMDN achieved higher likelihood than the others, random samples from the AR-RMDN were less natural than generating the mean trajectory. One reason may be that the AR model is still a weak model of the time dependency in the feature trajectory. Another reason may be that the dependency across dimensions of the feature vector is ignored [27]. In this case, the real-valued neural autoregressive density-estimator (RNADE) may be useful [28].

6. CONCLUSION

We introduced the autoregressive model into the recurrent mixture density network for speech synthesis. Experimental results showed that the trainable autoregressive model amplifies the high-frequency components and attenuates the low-frequency part of the target acoustic features in the training stage. In the synthesis stage, it compensates the low-frequency components and de-emphasizes the high-frequency components of raw trajectories generated by the neural network. Further analysis has shown that, even through the explicit global variance criterion is not used, the proposed model increases the level of the global variance of the generated feature trajectories because of the AR model in the synthesis stage. Our proposed model has achieved higher likelihood on the training and validation data, and better quality of synthetic speech.

7. ACKNOWLEDGEMENT

We are indebted to Dr. Gustav Eje Henter for suggestions on this work and comments on this paper. The work presented in this paper was partially supported by EPSRC EP/J002526/1 (CAF), by the Core Research for Evolutional Science and Technology (CREST) from the Japan Science and Technology Agency (JST) (uDialogue project), by MEXT KAKENHI Grant Numbers (26280066, 26540092, 15H01686, 15K12071, 16H06302, 16K16096), and by The Telecommunications Advancement Foundation Grant.

8. REFERENCES

[1] Keiichi Tokuda, Yoshihiko Nankaku, Tomoki Toda, Heiga Zen, Junichi Yamagishi, and Keiichiro Oura, "Speech synthesis based on hidden Markov models," *Proceedings of the IEEE*, vol. 101, no. 5, pp. 1234–1252, 2013.

[2] Heiga Zen, Alan Senior, and Martin Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Proc. ICASSP*, 2013, pp. 7962–7966.

[3] Yuchen Fan, Yap Qian, Feilong Xie, and Frank K. Soong, "TTS synthesis with bidirectional LSTM based recurrent neural networks," *Proc.Interspeech*, pp. 1964–1968, 2014.

[4] Zhen-Hua Ling, Li Deng, and Dong Yu, "Modeling spectral envelopes using restricted Boltzmann machines and deep belief networks for statistical parametric speech synthesis," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 10, pp. 2129–2139, 2013.

[5] Shinji Takaki and Junichi Yamagishi, "A deep auto-encoder based low-dimensional feature extraction from FFT spectral envelopes for statistical parametric speech synthesis," in *Proc. ICASSP*, 2016, pp. 5535–5539.

[6] Christopher M. Bishop, "Mixture Density Networks," Tech. Rep., Aston University, 2004.

[7] Sophocles J. Orfanidis, *Optimum Signal Processing: An Introduction*, Macmillan publishing company, 1988.

[8] Matt Shannon, Heiga Zen, and William Byrne, "Autoregressive models for statistical parametric speech synthesis," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 3, pp. 587–597, 2013.

[9] Carl Quillen, "Autoregressive HMM speech synthesis," in *Proc. ICASSP*, 2012, pp. 4021–4024.

[10] Tomoki Toda and Keiichi Tokuda, "A speech parameter generation algorithm considering global variance for HMM-based speech synthesis," *IEICE Transactions on Information and Systems*, vol. 90, no. 5, pp. 816–824, 2007.

[11] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber, "Gradient flow in recurrent nets: The difficulty of learning longterm dependencies," in *A Field Guide to Dynamical Recurrent Networks*, 2001, pp. 237–243.

[12] Alex Graves, *Supervised Sequence Labelling with Recurrent Neural Networks*, Ph.D. thesis, Technische Universität München, 2008.

[13] Heiga Zen and Andrew Senior, "Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis," in *Proc. ICASSP*, 2014, pp. 3844–3848.

[14] Heiga Zen, Keiichi Tokuda, and Tadashi Kitamura, "Reformulating the HMM as a trajectory model by imposing explicit relationships between static and dynamic feature vector sequences," *Computer Speech & Language*, vol. 21, no. 1, pp. 153–173, 2007.

[15] Kei Hashimoto, Keiichiro Oura, Yoshihiko Nankaku, and Keiichi Tokuda, "Trajectory training considering global variance for speech synthesis based on neural networks," in *Proc. ICASSP*, 2016, pp. 5600–5604.

[16] Matt Shannon and William Byrne, "A formulation of the autoregressive HMM for speech synthesis," Tech. Rep., University of Cambridge, CUED/F-INFENG/TR.629, 2009.

[17] Heiga Zen and Haşim Sak, "Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis," in *Proc. ICASSP*, 2015, pp. 4470–4474.

[18] Simon King and Vasilis Karaiskos, "The Blizzard Challenge 2011," in *Proc. Blizzard Challenge 2011*, 2011.

[19] Hideki Kawahara, Ikuyo Masuda-Katsuse, and Alain de Cheveigne, "Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds," *Speech Communication*, vol. 27, pp. 187–207, 1999.

[20] HTS Working Group, "The English TTS System 'Flite+HTS_engine'," 2014.

[21] Felix Weninger, Johannes Bergmann, and Björn Schuller, "Introducing CURRENT: The Munich open-source CUDA recurrent neural network toolkit," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 547–551, 2015.

[22] John D. Markel, "Digital inverse filtering-A new tool for formant trajectory estimation," *IEEE Transactions on Audio and Electroacoustics*, vol. 20, no. 2, pp. 129–137, Jun 1972.

[23] Mary E. Beckman and Gayle Ayers, "Guidelines for ToBI labelling," *The OSU Research Foundation*, vol. 3, 1997.

[24] Gustav E. Henter, Srikanth Ronanki, Oliver Watts, Mirjam Wester, Zhizheng Wu, and Simon King, "Robust TTS duration modelling using DNNs," in *Proc. ICASSP*, 2016, pp. 5130–5134.

[25] Heiga Zen, Yannis Agiomyrgiannakis, Niels Egberts, Fergus Henderson, and Przemyslaw Szczepaniak, "Fast, compact, and high quality LSTM-RNN based statistical parametric speech synthesizers for mobile devices," *CoRR*, vol. abs/1606.06061, 2016.

[26] Shinnosuke Takamichi, Tomoki Toda, Graham Neubig, Sakriani Sakti, and Satoshi Nakamura, "A postfilter to modify the modulation spectrum in HMM-based speech synthesis," in *Proc. ICASSP*, 2014, pp. 290–294.

[27] Gustav E. Henter, Thomas Merritt, Matt Shannon, Catherine Mayo, and Simon King, "Measuring the perceptual effects of modelling assumptions in speech synthesis using stimuli constructed from repeated natural speech.," in *Proc. Interspeech*, 2014, pp. 1504–1508.

[28] Benigno Uria, Iain Murray, Steve Renals, Cassia Valentini-Botinhao, and John Bridle, "Modelling acoustic feature dependencies with artificial neural networks: Trajectory-RNADE," in *Proc. ICASSP*, 2015, pp. 4465–4469.